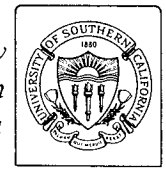
*ISI Reprint Series*
*ISI/RS-90-253*
*April 1990*






Richard J. Lipton
David W. Mizell

# Time Warp vs. Chandy–Misra: A Worst–Case Comparison

Reprinted from *Distributed Simulation*,
the proceedings of the SCS Multiconference
on Distributed Simulation, held 17–19 January 1990
in San Diego, California.

*INFORMATION SCIENCES INSTITUTE*

*213/822-1511*
*4676 Admiralty Way/Marina del Rey/California 90292-6695*

*ISI Reprint Series*
*ISI/RS-90-253*
*April 1990*

*University of Southern California*

Richard J. Lipton
David W. Mizell

# Time Warp vs. Chandy–Misra: A Worst–Case Comparison

Reprinted from *Distributed Simulation*, the proceedings of the SCS Multiconference on Distributed Simulation, held 17–19 January 1990 in San Diego, California.

*INFORMATION SCIENCES INSTITUTE*

*213/822-1511*

*4676 Admiralty Way/Marina del Rey/California 90292-6695*

This research was sponsored in part by the Defense Advanced Research Projects Agency under the Office of Naval Research Contract Numbers N00014-85-C-0456 and N00014-85-K-0465, and under National Aeronautics Space Administration Cooperative Agreement NCC2-539, and in part by the National Science Foundation under Cooperative Agreement DCR-8420948. Views and conclusions contained in this report are the authors' and should not be interpreted as representing the official opinion or policy of DARPA, ONR, NASA, NSF, the U.S. Government, or any person or agency connected with them.

# Time Warp vs. Chandy-Misra: A Worst-Case Comparison

Richard J. Lipton
Department of Computer Science
Princeton University
Princeton, NJ 08544*

David W. Mizell
Boeing Computer Services
P.O. Box 24346
Seattle, WA 98124-0346 †

## Abstract

We address the question of whether one can find a worst-case example simulation model, on which the Time Warp approach to parallel discrete event simulation can arbitrarily outperform the Chandy-Misra conservative methods – or vice versa.

Under our simplifying assumptions, we prove that (1) there exists a $p$-process simulation model on which Time Warp outperforms Chandy-Misra by a factor of $p$, and that (2) no opposite example exists; Chandy-Misra can only outperform Time Warp by a constant factor.

## 1 Introduction

Two general methods are advocated for assuring correct distributed simulation: the *conservative* approach, commonly associated with Chandy and Misra [1, 2, 3, 4, 5, 6, 7] and the *optimistic* or "Time Warp" method, invented by Jefferson and Sowizral [8, 9]. There have been several studies of distributed simulation methods, some experimental and some analytical. Seethalakshmi's thesis [10] reported some simulation studies of a Chandy-Misra method. Reed and Fujimoto have conducted experimental studies on conservative methods, as well [11, 12]. Experimental studies of the Time Warp approach have been carried out by Berry [13], Jefferson *et al* [14], and Fujimoto [15]. Fairly narrowly focused analytical studies of Time Warp have been reported by Lavenberg *et al* [16] and Jefferson and Witkowski [17].

Little experimental or analytical work has directly compared the conservative and optimistic approaches, with notable exceptions of some very recent work by Lin and Lazowska [18] and by Bailey and Snyder [19]. Lin and Lazowska prove that Time Warp outperforms Chandy-Misra on feedforward networks and in most cases of feedback networks without lookahead. Bailey and Snyder give an example which illustrates that, if the ratio of simulation processes to processors is greater than one, then either strategy can outperform the other, depending on how event processing is scheduled.

It is our perception that the possible behaviors of each of the distributed simulation approaches are so complex that the number of assumptions necessary to make a general analysis tractable may also make it meaningless. Instead, we addressed a narrower, worst-case question: Can one find an example simulation on which Time Warp would *arbitrarily outperform* Chandy-Misra, or vice versa? By "arbitrarily outperform" we mean the following: Simulation method A arbitrarily outperforms simulation method B on a given, $p$-process simulation if the method A simulator is proportional to $p$ times faster than the method B simulator.

We show an (artificial) example simulation for which the Time Warp implementation arbitrarily outperforms the Chandy-Misra implementation; on the other hand, we prove that under our assumptions, Chandy-Misra cannot arbitrarily outperform Time Warp.

*Supported in part by DARPA and ONR contracts N00014-85-C-0456 and N00014-85-K-0465, and by NSF Cooperative Agreement DCR-8420948.

†Supported in part by SDIO via DARPA and NASA, by NASA Cooperative Agreement NCC2-539, at USC/ISI.

## 2 Assumptions

First, we establish the context of these results by stating the assumptions we made about the way the Time Warp and Chandy-Misra simulation systems are implemented, and the multiprocessor on which each is to execute.

- There is an arbitrary number of processors available.
- We will only consider simulation *processing* time in this analysis. We assume that sending and receiving messages costs the processor no compute time.
- It costs a processor a fixed amount of execution time $t_e \geq 1$ to compute a new event.
- Each Time Warp process saves state immediately following every event it computes.
- It costs a processor a fixed amount of execution time $d_r \geq 1$ to roll back a Time Warp process.
- If a Time Warp process encounters two or more messages in its input queue at the same time, each of which would cause a rollback, it processes the earliest-timestamped of these first. Also, if a Time Warp process encounters a message and its antimessage together in its input queue, its costs the process no time to cancel them. In other words, a Time Warp process pays the rollback cost only for the "tardiest" of the messages in its input queue at a given time.
- We assume that a Chandy-Misra process informs every process it can directly affect every time it changes its local simulation clock, and that it costs a Chandy-Misra process unit time to update its simulation clock based on non-event-related messages, such as null messages [1, 3], appointments [20], etc.
- Newly arriving messages don't interrupt a processor; it will finish the currently ongoing event, rollback, or clock update operation before examining a message that arrives during that operation.
- The cost of deciding that the simulation has terminated is ignored.

## 3 The Comparison Results

In previous experimental studies of distributed simulation, specific physical systems (pool balls undergoing elastic collision seems to have been a favorite) were used as examples. In their analytical study [19], Bailey and Snyder focus upon digital circuits as the simulation "subjects." For this work, in particular for the existence proof in Theorem 1, we needed to specify artificial systems with arbitrary event dependency characteristics. To this end, we invented an algebraic construct we call the "simulation model."

Details of the abstraction will be found in a companion paper [21]; we will sketch here only what is necessary to make sense of the example. Each "physical process" [7] of the system being simulated is represented by a particular kind of finite-state machine that we call a *timed state machine*. Each machine is in its distinguished starting state when the simulation begins. An *event* occurs at an instant in time when a machine changes state. There are two types of such events. The first is called a *timeout* event. If a state machine has been in a particular state for a fixed maximum time associated with that state, it changes to a designated next state at the end of that interval. The second type of event is a *caused* event. A timeout event in one machine can cause another machine to change state at the same instant, to a state not necessarily the same as the one it would have changed to as its timeout event.

There is a relation between simulated events sometimes used in discussions of distributed simulation. For convenience, we will treat it as a directed graph. The *event history* is a graph in which every node is an event computed by the simulator. The edges between nodes represent a partial ordering based on increasing simulation time and the following rules:

1. If event $e_i$ caused event $e_j$ in another state machine, there is an edge from $e_i$ to $e_j$.

2. If $e_j$ immediately succeeds $e_i$ in the same state machine and $e_j$ is a timeout event, there is an edge from $e_i$ to $e_j$.

**Theorem 1:** *There exists a simulation model consisting of p timed state machines for which a Time Warp implementation on a p-node multiprocessor outperforms a Chandy-Misra implementation on the same multiprocessor by a factor of p.*

**Proof:** We prove by constructing such an example simulation model. Intuitively, the simulation model is one such that Time Warp can complete almost all of the simulation in parallel by "guessing right" while a Chandy-Misra implementation is constrained to proceed no faster than a sequential implementation. The example is shown in a pseudocode representation of the timed state machines below.

Physical Process 1:

State $1_{START}$: at time $\frac{1}{2}$, flip a coin.
if heads, change to State $1_A$;
else change to State $1_B$.
State $1_A$: stay in this state for 1 time unit,
then change to State $1_D$.
State $1_B$: stay in this state for 1 time unit,
then change to State $1_C$
State $1_C$: dead
State $1_D$: dead

Physical Process 2:

State $2_{START}$: change to State $2_A$ at time 2
unless event $1_B \rightarrow 1_C$ occurs in
Physical Process 1 earlier, in which case
change to state $2_B$.
State $2_A$: dead
State $2_B$: stay in this state for 1 time unit,
then change to State $2_C$
State $2_C$: dead

Physical Process 3:

State $3_{START}$: change to State $3_A$ at time 3
unless event $2_B \rightarrow 2_C$ occurs in
Physical Process 2 earlier, in which case
change to state $3_B$.
State $3_A$: dead
State $3_B$: stay in this state for 1 time unit,
then change to State $3_C$
State $3_C$: dead

Physical Processes 4 through $p$: similar to Physical Processes 2 and 3.

Consider the behavior of a Chandy-Misra implementation of this simulation model. PP1 will affect PP2 at time 1.5 if the coin flip comes out heads but will never affect PP2 otherwise. Since the timeout event of PP2's starting state is not until time 2, PP2 can do nothing before PP1 informs it that it is in a dead state. Similarly, PP3 must wait for PP2, and so on. The example is constructed in such a way that techniques based on lookahead [3, 22, 20] or conditional events [23] will not speed up a Chandy-Misra implementation: process $i$'s next event time is earlier than process $i+1$'s timeout, so advance knowledge of this time does not enable process $i+1$ to proceed. Thus, any Chandy-Misra implementation must proceed sequentially on this simulation model.

In the Time Warp implementation, on the other hand, each Time Warp process will immediately execute the timeout event for the starting state of the physical process it represents. If the coin flip comes out heads, the Time Warp processes will have all guessed correctly, and simultaneously computed the correct event for their physical process.

This result is probably consistent with the intuition of most readers, that it is possible for Time Warp to "win big" over a Chandy-Misra implementation if it is lucky enough to always guess correctly whenever it is basing its computations on incomplete information. Of course, the existence-proof nature of Theorem 1 sheds no light at all on the likelihood of correct guesses in general.

The natural follow-up question to ask was whether or not an opposite example existed, on which a Chandy-Misra implementation would achieve linear speedup and a Time Warp implementation would make progress no faster than a se-

quential implementation. The answer turns out to be that no such example exists; under our assumptions, Time Warp will never execute more slowly than a constant times the Chandy-Misra execution time. The intuition behind this result is that trailing along behind every Time Warp simulation like a shadow is another simulation that behaves very much like a Chandy-Misra implementation, and this "shadow" simulation cannot lag more than a constant factor behind a real Chandy-Misra simulation.

**Definition:** *We say that a Time Warp process* casts *an event when it computes the event for the last time in a simulation run.*

As it rolls back and then moves forward again, it is possible for a Time Warp process to compute the same event several times – but it will only cast the event once. Note that we make no assumption that the Time Warp process knows when it casts an event – typically it does not find out that its earlier-computed events are cast until the next time that global virtual time [9] is computed – we simply point out that the last computation of an event will occur. Note also that every event that is cast is a valid event rather than the consequence of a wrong guess (since we assume a correct implementation of Time Warp) and that events are cast in strictly increasing timestamp order within each process.

**Definition:** *Invisible to each Time Warp process we postulate the existence of a* shadow clock *for that process, which always holds the simulation clock time of the latest cast event.*

Note that throughout a Time Warp simulation, global virtual time is equal to the minimum across all processes of the shadow clock values.

These hypothetical shadow clocks move strictly forward in a manner quite similar to the real local simulation time clocks in the Chandy-Misra processes, although not necessarily at the same rate. The following lemma establishes a bound on how much more slowly the shadow clocks can advance than the Chandy-Misra clocks, and illustrates the similarity in behavior between the two.

**Lemma:** *Suppose that at some point during a Time Warp simulation, the following conditions hold:*

1. *process $TW_i$ has its shadow clock at time $t$*
2. *the next event that $TW_i$ will cast will be at simulation time $t' \geq t$*
3. *all other simulation processes that can directly affect $TW_i$ (i.e., those that have output edges going to $TW_i$ in the event dependency graph) have their shadow clocks at times greater than $t'$.*

*Then $TW_i$ will cast the next event after an amount of processing time no greater than $d_r + max\{d_r, t_e\} + t_e$.*

**Proof:** The proof consists of three points:

1. $TW_i$ computes the correct event. This is so because all the processes which can affect $TW_i$ have their shadow clocks beyond $t'$. Therefore they are correct up to points beyond $t'$. Therefore $TW_i$ has all the correct information it needs to compute the next valid event.

2. $TW_i$ will not only compute the next event, it will cast it. This is for the same reason as above: because the shadow clocks of all the processes that can affect $TW_i$ are beyond $t'$, nothing can ever cause $TW_i$ to roll back to a simulation time earlier than $t'$. Note that this point depends on our assumption that every Time Warp process saves state after every event it computes.

3. The terms of the expression of maximum processing cost are based on the following:

   $d_r$: the arrival of the last of the correct information from the processes that can affect $TW_i$ may cause $TW_i$ to roll back, if it had been proceeding incorrectly based on incomplete information.

   $max\{d_r, t_e\}$: this term is a consequence of the assumption that newly arriving messages do not interrupt a Time Warp process. Thus, $TW_i$ may be performing a rollback or an event computation at the time the last of the correct information arrives, and it will complete this action before checking the input queue.

   $t_e$: this is the time required to compute (cast) the valid event.

This lemma illustrates the sense in which a Time Warp simulation, as it casts events and in so doing advances simulation time on the imaginary shadow clocks, mimics the behavior of a Chandy-Misra simulation. The situation in which a Time Warp process's input processes have their shadow clocks ahead of the time of the next event it will cast corresponds closely to the situation in which a Chandy-Misra process is able to compute its next event.

Furthermore, the lemma establishes an upper bound on the time it takes shadow clock information to propagate across processes and enable other shadow clocks to advance: for any path through the event dependency graph, it can take as long as the expression in the lemma, multiplied by the number of edges in the path.

Note that a lower bound on the rate at which clock information propagates in a Chandy-Misra simulation is simply the number of edges in the path, since we assume that it takes a Chandy-Misra process unit time to update its simulation clock. These bounds on the difference between the fastest rate of Chandy-Misra simulation clock information propagation and the slowest possible rate for Time Warp shadow clock information are key to Theorem 2.

**Theorem 2:** *If a Chandy-Misra implementation of a simulation model can compute an event history graph of maximum depth $k$ in processing time $T$, then a Time Warp implementation of the simulation model can cast the same events in processing time no greater than* $(d_r + max\{d_r, t_e\} + t_e)T$.

**Proof** is by induction on $k$.

**k=1:** Each of the events computed must be a timeout event for the process that computes it. This process must either be a source process or a member of a cycle; if it were a chain process, a history of depth greater than 1 would have to be computed. If it is a source process, both Time Warp and Chandy-Misra compute it in time $t_e$. If it is in a cycle, Time Warp casts the event in time $t_e$ while Chandy-Misra takes at least $c + t_e$, where $c$ is the number of processes in the cycle.

**assume for $k \leq N$ and prove for $k = N + 1$:** Let the time the Chandy-Misra simulation took to compute an event history of depth $N$ be $T - t_e - \Delta$. By the induction hypothesis, Time Warp cast the events in time no greater than $(d_r + max\{d_r, t_e\} + t_e)(T - t_e - \Delta)$. The Chandy-Misra simulator propagated the necessary clock information in time $\Delta$, and then computed the final event or events in time $t_e$. By the lemma, Time Warp could propagate the corresponding shadow clock information in time no greater than $(d_r + max\{d_r, t_e\} + t_e)\Delta$ and cast the final event(s) in time $d_r + max\{d_r, t_e\} + t_e$. Thus, the total time for the Time Warp simulation would be bounded above by $(d_r + max\{d_r, t_e\} + t_e)(T - t_e + 1)$, which, since $t_e \geq 1$, is bounded above by $(d_r + max\{d_r, t_e\} + t_e)T$.

It is fairly easy to construct an example simulation model for which the Chandy-Misra implementation outperforms Time Warp by the factor given in the theorem. The example is intriguingly similar to the one used in Theorem 1 – the event dependency graph is again a chain – only this time, Time Warp always guesses incorrectly, and must roll back at every step while Chandy-Misra makes progress.

# 4 Conclusions

Theorem 1 reinforces a common intuition about Time Warp vs. Chandy-Misra: that Time Warp can "win big" if it is lucky enough to consistently guess right. Theorem 2 points out that, on the other hand, Time Warp cannot lag arbitrarily far behind Chandy-Misra, because within a constant factor of the time a Chandy-Misra process receives the information that enables it to make progress, a Time Warp process will receive what amounts to the same information, causing it to correct itself and then make progress.

Of course, more general comparisons of the distributed simulation methods under more realistic assumptions, especially with respect to message traffic, would be more useful than these results. We plan to use the simulation model construct as a design basis for an experimental investigation into how the computation and communication require-

ments imposed by the structure of the entity being simulated affect the performance of both approaches.

# 5 Acknowledgements

Credit for the original suggestion of this research topic goes to Skip Saunders. The authors benefited greatly from discussions of distributed simulation with Rivi Sherman, Walid Najjar and Henry Sowizral. Rivi, in particular, suggested the basic structure of the example in Theorem 1. The second author also wishes to thank Bob Brown and Jon Postel of USC/ISI for arranging a work situation in which he could concentrate on this research.

# References

[1] R.E. Bryant. Simulation of packet communication architecture computer systems. Technical Report TR-188, M.I.T. Laboratory for Computer Science, 1977.

[2] K.M. Chandy, J. Misra, and V. Holmes. Distributed simulation of networks. *Computer Networks*, 3:105–113, 1979.

[3] K.M. Chandy and J. Misra. Distributed simulation: A case study in design and verification of distributed programs. *IEEE Transactions on Software Engineering*, SE-5(5):440–452, 1979.

[4] J.K. Peacock, W. Wong, and E. Manning. A distributed approach to queueing network simulation. In *Proceedings, IEEE 1979 Winter Simulation Conference*, pages 399–406, 1979.

[5] J.K. Peacock, W. Wong, and E. Manning. Distributed simulation using a network of processors. *Computer Networks*, pages 44–56, February 1979.

[6] K.M. Chandy and J. Misra. Asynchronous distributed simulation via a sequence of parallel computations. *Communications of the ACM*, 24(11):198–206, April 1981.

[7] J. Misra. Distributed discrete-event simulation. *ACM Computing Surveys*, 18(1):39–65, March 1986.

[8] David Jefferson and Henry Sowizral. Fast concurrent simulation using the Time Warp mechanism, Part I: Local control. Technical Report N-1906-AF, Rand Corporation, December 1982.

[9] D. Jefferson and H. Sowizral. Fast concurrent simulation using the Time Warp mechanism. In *Proceedings, SCS Distributed Simulation Conference*, January 1985.

[10] M. Seethalakshmi. A study and analysis of performance of distributed simulation. Master's thesis, University of Texas, 1979.

[11] Daniel A. Reed and Allen D. Malony. Parallel discrete event simulation: the Chandy-Misra approach. In *Proceedings of the Conference on Distributed Simulation*, San Diego, 1987. Society of Computer Simulation International.

[12] Richard Fujimoto. Performance measurements of distributed simulation strategies. In *Proceedings of the Conference on Distributed Simulation*, San Diego, 1987. Society of Computer Simulation International.

[13] Orna Berry. *Performance Evaluation of the Time Warp Distributed Simulation Mechanism.* PhD thesis, University of Southern California, May 1986.

[14] David Jefferson, Brian Beckman, Steve Hughes, Eric Levy, Todd Litwin, John Spagnuolo, Jon Vavrus, Fred Wieland, and Barbara Zimmerman. Implementation of Time Warp on the Caltech hypercube. In *Proceedings of the Conference on Distributed Simulation*, San Diego, January 1985.

[15] Richard Fujimoto. Time Warp on a shared memory multiprocessor. Technical Report UUCS-88-021a, University of Utah, 1989.

[16] S. Lavenberg, R. Muntz, and B. Samadi. Performance analysis of a rollback method for dis-

tributed simulation. In $9^{th}$ *International Symposium on Computer Performance Modeling, Measurement, and Evaluation*, pages 117–132, May 1983.

[17] David Jefferson and Andrew Witkowski. An approach to performance analysis of timestamp-driven synchronization mechanisms. In *Proceedings of the 3rd Annual ACM Conference on Distributed Computing*, 1984.

[18] Yi-Bing Lin and Edward J. Lazowska. Optimality considerations for "Time Warp" parallel simulation. Technical Report 89-07-05, University of Washington Computer Science Department, July 1989.

[19] Mary L. Bailey and Lawrence Snyder. A model for comparing synchronization strategies for parallel logic-level simulation. In *Proceedings of the International Conference on Computer-Aided Design*, 1989. To appear.

[20] David M. Nicol. Parallel discrete-event simulation of FCFS stochastic queueing networks. In *ACM/SIGPLAN Parallel Programming: Experience with Applications, Languages and Systems*, pages 124–137, New Haven, CT, July 1988.

[21] David W. Mizell and Richard J. Lipton. Worst-case comparison of optimistic and conservative parallel discrete-event simulation methods. Technical report, USC Information Sciences Institute, 1989. In preparation.

[22] B.D. Lubachevsky. Bounded lag distributed discrete event simulation. In *1988 SCS Conference on Distributed Simulation*, pages 183–191, San Diego, CA, 1988.

[23] K.M. Chandy and Rivi Sherman. The conditional-event approach to distributed simulation. In *SCS Conference on Distributed Simulation*, Tampa, Florida, March 1989.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

# REPORT DOCUMENTATION PAGE

| | |
|---|---|
| 1a. REPORT SECURITY CLASSIFICATION<br>Unclassified | 1b. RESTRICTIVE MARKINGS |
| 2a. SECURITY CLASSIFICATION AUTHORITY | 3. DISTRIBUTION / AVAILABILITY OF REPORT<br>This document is approved for public release; distribution is unlimited. |
| 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE | |
| 4. PERFORMING ORGANIZATION REPORT NUMBER(S)<br>ISI/RS-90-253 | 5. MONITORING ORGANIZATION REPORT NUMBER(S)<br>------- |
| 6a. NAME OF PERFORMING ORGANIZATION<br>USC/Information Sciences Institute<br>6b. OFFICE SYMBOL *(If applicable)* | 7a. NAME OF MONITORING ORGANIZATION<br>ONR, NASA-Ames |
| 6c. ADDRESS *(City, State, and ZIP Code)*<br>4676 Admiralty Way<br>Marina del Rey, CA 90292-6695 | 7b. ADDRESS *(City, State, and ZIP Code)*<br>--over-- |
| 8a. NAME OF FUNDING / SPONSORING ORGANIZATION<br>DARPA, NSF<br>8b. OFFICE SYMBOL *(If applicable)* | 9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER<br>--over-- |
| 8c. ADDRESS *(City, State, and ZIP Code)*<br>--over-- | 10. SOURCE OF FUNDING NUMBERS<br>PROGRAM ELEMENT NO. ------- / PROJECT NO. ------- / TASK NO. ------- / WORK UNIT ACCESSION NO. ------- |

11. TITLE *(Include Security Classification)*
Time Warp vs. Chandy-Misra: A Worst-Case Comparison (Unclassified)

12. PERSONAL AUTHOR(S) Lipton, Richard J.; Mizell, David W.

| 13a. TYPE OF REPORT | 13b. TIME COVERED | 14. DATE OF REPORT *(Year, Month, Day)* | 15. PAGE COUNT |
|---|---|---|---|
| Research Report | FROM _____ TO _____ | 1990, April | 12 |

16. SUPPLEMENTARY NOTATION
Reprinted from *Distributed Simulation*, the proceedings of the SCS Multiconference on Distributed Simulation (David Nicol, ed.), held 17-19 January 1990 in San Diego, California.

| 17. COSATI CODES | | | 18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)* |
|---|---|---|---|
| FIELD | GROUP | SUB-GROUP | distributed simulation |
| 09 | 02 | | |
| | | | |

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

We address the question of whether one can find a worst-case example simulation model, on which the Time Warp approach to parallel discrete event simulation can arbitrarily outperform the Chandy-Misra conservative methods -- or vice versa. Under our simplifying assumptions, we prove that (1) there exists a $p$-process simulation model on which Time Warp outperforms Chandy-Misra by a factor of $p$, and that (2) no opposite example exists; Chandy-Misra can only outperform Time Warp by a constant factor.

| 20. DISTRIBUTION / AVAILABILITY OF ABSTRACT | 21. ABSTRACT SECURITY CLASSIFICATION |
|---|---|
| ☒ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS | Unclassified |

| 22a. NAME OF RESPONSIBLE INDIVIDUAL | 22b. TELEPHONE *(Include Area Code)* | 22c. OFFICE SYMBOL |
|---|---|---|
| Victor Brown Sheila Coyazo | 213/822-1511 | |

**7b. continued**

Office of Naval Research
UC San Diego
Scripps Institute (A-034)
8603 La Jolla Shores Dr.
San Diego, CA 92092-0234

NASA-Ames
Moffett Field, CA 94035

**8c. continued**

Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, VA 22209

National Science Foundation
1800 G Street NW
Washington, DC 20550

**9. continued**

DARPA (ONR): N00014-85-C-0456
N00014-85-K-0465

DARPA (NASA-Ames): NCC-2-539

NSF: DCR-8420948